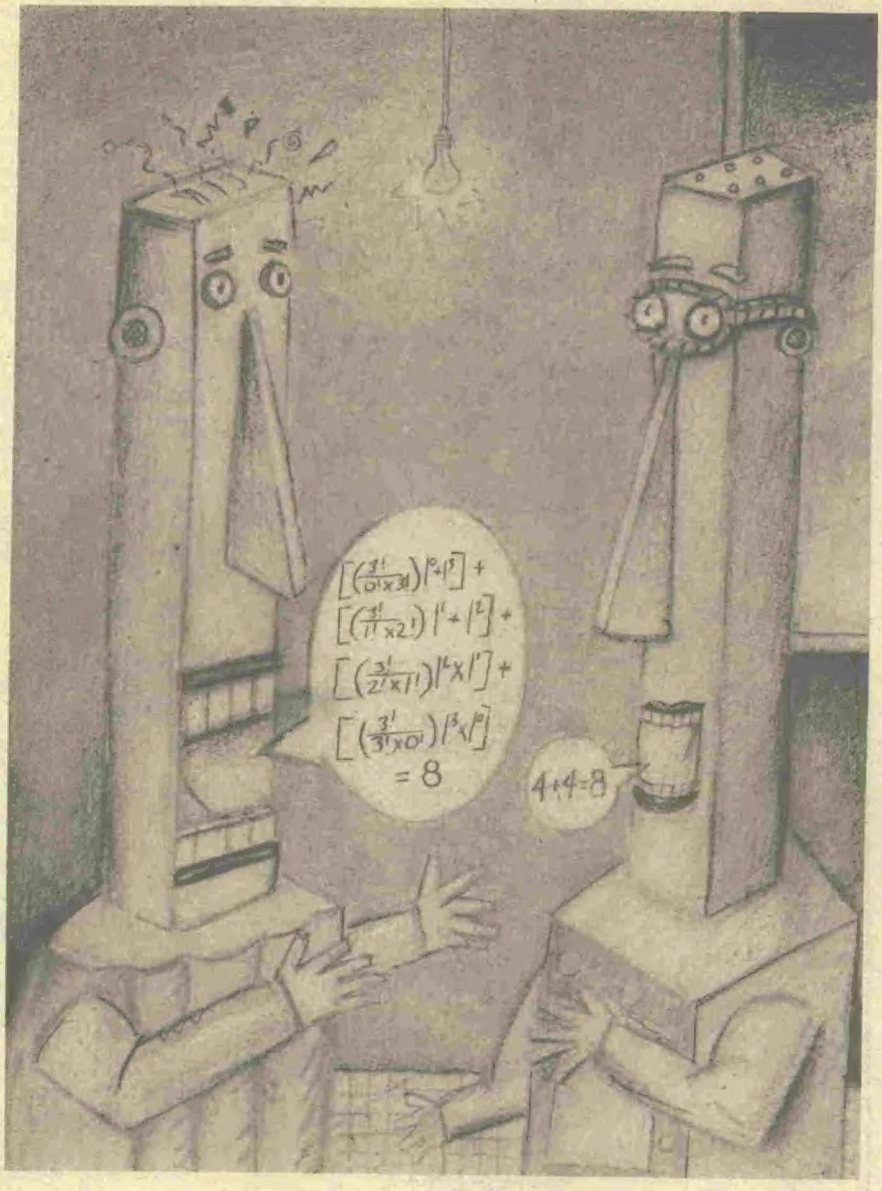
กลยุทธ์การเขียน โปรแกรมแบบขนาน

จุ๊...จุ๊...เคล็ดลับสุดยอดพิเศษในการเขียนโปรแกรมแบบขนานที่ว่ายุ่งนักยุ่งหนาขนาดไหน ให้กลายเป็นเรื่องง่าย ๆ สบาย ๆ



มือเครื่องคอมพิวเตอร์ที่ทำงานขนาน (Parallel computers) เริ่มแพร่หลายและราคาถูกลง โปรแกรมเมอร์สมองใสใจไม่หยุดนิ่งก็เริ่มมองหาวิธีแปลงกายโปรแกรมโบราณให้ไปใช้อาวุธใหม่! ทั้งในแวดวงวิทยาศาสตร์ วิศวกรรมศาสตร์ ระบาดไปจนถึงวงการธุรกิจ ปัญหาที่ต้องใช้งานเกี่ยวกับการคำนวณสูง ๆ ยิ่งมายิ่งมากยิ่งสาหัส ถึงแม้เครื่องคอมพิวเตอร์ยุคใหม่จะทำงานเร็วขึ้น แต่ก็มีขีดที่สิ้นสุด เนื่องจากสภาวะนิสัยที่ทำงานเป็นขั้นเป็นตอนเรียงลำดับของตัวคอมพิวเตอร์เอง ทำให้ไม่สามารถปั่นงานออกมาปริมาณมาก ๆ ได้ เครื่องคอมพิวเตอร์ชนิดทำงานขนานอาศัยหลักการหลายหัวดีกว่าหัวเดียว ใช้หน่วยประมวลผลหลายตัวช่วยกันรุมสกรัมปัญหาชิ้นเดียว อย่างนี้จะไม่ให้งานเสร็จเร็วอย่างไรไหว

แต่ก่อนกาลนานมา ถ้าบริษัทไม่ใหญ่จริงเรื่องที่จะทุ่มทุนถึง 1,800 ล้านบาท (ชะเมื่อไร) ซื้อหาเจ้า Parallel computer มาใช้นั้นคงไม่มีทาง เมื่อเทคโนโลยีก้าวหน้ายิ่งขึ้น ราคาเครื่องค่อย ๆ ขยับลดลง ก็เริ่มมองเห็นแววความคุ้มค่าน่าลงทุน เพราะ Parallel computer ตัวหนึ่งอาจจะมีขีดความสามารถเทียบเท่าซูเปอร์คอมพิวเตอร์ แต่พอลองเทียบราคาแล้วอาจกล่าวได้ว่า *"มันช่างจิ๊บจ๊อยเหลือหลาย"*

ฤาจะแปลงกายสู่เจ้าให้เข้าที

ปัญหามีอยู่ว่า ถ้าไม่มีโปรแกรมสนับสนุนที่ดีพอ งานของคุณก็ไม่สามารถใช้งานความเป็นคอมพิวเตอร์

ขนานได้อย่างเต็มที่ ทีนี้ถ้ามีกรรมวิธีที่จะแปลงโปรแกรมเก่า ๆ ที่เขียนสำหรับใช้งานกับคอมพิวเตอร์ธรรมดา ๆ มาใช้งานกับเครื่องยุคใหม่ได้ก็จะดีไม่น้อย เผอิญมีกรรมวิธีอย่างหนึ่งเรียกว่า LINDA สามารถทำอย่างที่ว่าได้พอดี

สาเหตุที่การประมวลผลแบบขนานถูกทอดทิ้งไม่ใส่ใจนั้นส่วนหนึ่งมาจากภาษาที่ใช้ในการเขียนโปรแกรมให้ใช้กับเครื่องเหล่านี้นั้นผูกพันกับตัวเครื่องเหนียวแน่นเกินไป บริษัทใครบริษัทมัน ต่างคนก็ต่างออกแบบเครื่องออกมาประหลาด ๆ แล้วก็ออกภาษาตามมาให้ใช้กับเครื่องของตน ด้วยเหตุนี้โปรแกรมที่เขียนบนเครื่องยี่ห้อหนึ่งจะไม่สามารถส่งถ่ายไปใช้กับเครื่องยี่ห้ออื่น ผลคือต้องเขียนใหม่สถานเดียว ไม่เหมือนโปรแกรมภาษาซี ปาสคาล หรือเบสิกเพียงแก้นิดเดียว (บางทีไม่ต้องแก้เลย) จะนำไปใช้งานกับเครื่องยี่ห้อไหนรุ่นไหนได้หมด ดังนั้นไม่ว่าใครจะแปลงโปรแกรมเก่ามาเป็นโปรแกรมสำหรับเครื่องใหม่ก็ต้องคอยพะวงว่าจะใช้ได้อีกนานแค่ไหน แล้วเครื่องที่ตัดสินใจเลือกซื้อมาจะรุ่งหรือร่วง เลยพาลปวดหัวล้มเลิกการแปลงโปรแกรมไปเลยรู้แล้วรู้รอด

ถ้าจะแบ่งเครื่องคอมพิวเตอร์ขนานยุคปัจจุบันพอจะได้เป็น 2 ประเภทใหญ่ ๆ คือ พวกหนึ่งใช้กรรมวิธีส่งข้อมูลข่าวสารหากัน (message passing) อีกพวกใช้ส่วนความจำร่วมกัน (shared memory) แต่ละพวกต่างก็มีปมเด่นปมด้อยของตนเอง ระบบส่วนความจำร่วมถือกันว่าเขียนโปรแกรมง่าย ใครอยากอ่านอยากเขียนข้อมูลตรงไหนก็ทำได้ตรง ๆ แต่มักประสบปัญหาตอนแย่งกันเขียนแย่งกันอ่าน อาจทำให้ข้อมูลสับสนได้ จึงต้องเตรียมการระมัดระวังป้องกันเป็นพิเศษ ในขณะที่ระบบส่งผ่านข่าวสารนั้นข้อมูลจะถูกส่งไปมาระหว่างหน่วยงาน (process) เท่านั้น ซึ่งช่วยลดความสับสนของข้อมูลลงได้ แต่ต้องพบอุปสรรคในการ "แต่งสาร" ที่จะส่งนั่นแหละ

ในเมื่อความแตกต่างมีดังนี้ โปรแกรมที่เขียนเพื่อเครื่องจำพวกหนึ่งย่อมแตกต่างจากโปรแกรมที่เขียนสำหรับใช้กับเครื่องอีกจำพวกหนึ่งอย่างแน่นอน แม้ว่าจะใช้ทำงานแบบเดียวกันก็เถอะ โชคดีที่มี LINDA ซึ่งทำงานได้กับเครื่องทั้งสองประเภท ทำให้สามารถจะถ่ายโปรแกรมจากเครื่องต่างกรรมวิธีการทำงานได้ ทั้งนี้เพราะภาษารุ่นใหม่นี้จะบดบังความแตกต่างของตัวเครื่องจักรออกไปให้ดูเหมือนกัน ฉะนั้นโปรแกรมที่เขียนด้วย LINDA จึงสามารถใช้งานกับเครื่องได้ทั้งสองชนิด ถึงตอนนี้บรรดาโปรแกรมเมอร์และเจ้าของกิจการทั้งหลายคงเริ่มสบายใจได้แล้ว เนื่องจากคุณสามารถใช้ภาษาเพียงภาษาเดียวอาละวาดไปได้ทั่วทุกเครื่องตั้งแต่ปัจจุบันยันอนาคตทีเดียว

ถ้าสังเกตให้ดีจะพบว่า แม้แต่ในโปรแกรมรุ่นเก่าสำหรับเครื่องรุ่นเก๋าส์ก็จะมีความเป็นคู่ขนาน (Parallelism) แฝงอยู่ ซึ่งถ้าสามารถนำออกมาใช้ให้เป็นประโยชน์ได้ จะทำให้งานเสร็จเร็วยิ่งขึ้น วิธีหาความเป็นคู่ขนานนั้นให้พิจารณาจากข้อมูลผลลัพธ์แล้วไตร่ตรองว่า งานนี้สามารถแยกเป็นงานชิ้นย่อย ๆ ที่ขาดจากกันโดยสิ้นเชิง ต่างคนต่างคิดเองได้เป็นอิสระหรือไม่ ตัวอย่างเช่น ข้อมูลในตารางแต่ละช่องสามารถคำนวณแยกกันได้ หรือถ้าข้อมูลมีความสัมพันธ์เกี่ยวโยงกัน อย่างเช่น ตารางบัญชีซึ่งช่องหนึ่ง ๆ มีผลกระทบต่อช่องอื่น ถ้าลองพิจารณาดี ๆ ก็ยังสามารถคำนวณแยกจากกันได้ เพียงแต่ต้องระวังจังหวะการคำนวณก่อนหลังเล็กน้อยเท่านั้น ฉะนั้นถ้าเป็นโปรแกรมรุ่นเก่าให้หาตรงส่วนที่มีการทำงานวนรอบซ้อน ๆ กัน (nested loops) ส่วนใหญ่จะเป็นแหล่งระบุว่า บริเวณนี้แหละที่มีข้อมูลที่สามารถคำนวณโดยวิธีคู่ขนานได้

จอมยุทธในบู๊ลิ้ม (งานวิจัย) หลายคนต่างต้องเปลืองขมองขบคิดหาตัวแปลภาษา (Compiler) ชนิดที่พอเอาโปรแกรมเก่าใส่เข้าไปปุ๊บก็จะได้โปรแกรมใหม่ออกมาปั๊บ เป็นแบบที่มีการคำนวณคู่ขนานเลย จอมยุทธเหล่านั้นล้วนประสบกับความจริงข้อหนึ่งคือ ถ้าจะให้โปรแกรมใช้งานความเป็นเครื่องคอมพิวเตอร์ทำงานขนานให้เต็มที่นั้น ควรเริ่มตั้งแต่ตอนคิดแก้ปัญหา (Algorithm) กันเลยทีเดียว ที่แปลกไปกว่านั้นคือขั้นตอนการคิดแก้ปัญหาแบบขนาน (Parallel algorithm) มักได้เปรียบกว่าขั้นตอนการคิดแก้ปัญหาปกติ แม้ว่าจะใช้กับเครื่องเก่าหรือภาษาเก่าก็เถอะ ยิ่งใช้งานกับ LINDA ด้วยแล้ว การแก้ปัญหาหนึ่ง ๆ

สามารถเอาไปดัดแปลงใช้กับปัญหาอื่น ๆ ได้ซ้ำแล้วซ้ำอีก ยิ่งกำไรเข้าไปใหญ่

การเขียนโปรแกรมโดยวิธีขนานจะเป็นการคำนึงถึงผลได้ผลเสียระหว่างการคำนวณและการแลกเปลี่ยนข้อมูลเป็นหลัก ทางหนึ่งคือแบ่งงานให้ละเอียดยิบเป็นชิ้นเล็กชิ้นน้อย ซึ่งถ้าแบ่งละเอียดเกินไป เวลาและกำลังของเครื่องส่วนใหญ่กลับถูกใช้ไปในการติดต่อสื่อสารส่งข้อมูลระหว่างงานย่อยต่าง ๆ ทำให้เสียทรัพยากรโดยใช่เหตุ ในขณะที่อีกวิธีคือแบ่งงานเป็นชิ้นใหญ่ๆ ไม่ต้องมากชิ้น ซึ่งถ้าแบ่งเป็นชิ้นใหญ่เกินไปอาจเกิดปัญหาอีก เพราะงานย่อยชิ้นที่เสร็จก่อนต้องรอนานแสนนานกว่างานย่อยอีกชิ้นจะเสร็จ ฉะนั้นการเลือกขนาดชิ้นงานควรเลือกให้เหมาะสม (พูดง่ายทำยาก) LINDA กระโจนเข้ามาช่วยในเรื่องนี้ โดยการให้ผู้เขียนโปรแกรมกำหนดการแบ่งขนาดชิ้นงานให้ใหญ่เล็กได้

นอกจากนี้โปรแกรมแบบคู่ขนานต้องคำนึงถึงการรอจังหวะเหมาะ (Synchronization) ด้วย โปรแกรมที่ดีควรจะทำให้การรอจังหวะเหล่านี้เสียเวลาน้อยที่สุด เนื่องจากข้อมูลหรือปัญหาในชีวิตประจำวันมักจะเกี่ยวข้องสัมพันธ์กันอยู่ ทำให้การคำนวณก่อนและหลังมีผลต่อความถูกต้องของคำตอบ

ไม่ว่าจะเขียนโปรแกรมแบบขนานขึ้นมาใหม่ทั้งหมดหรือดัดแปลง (กาย) มาจากโปรแกรมเก่า หลักการก็ยังคงเดิม คือ วิธีที่ 1 ต้องหาทาง "หั่น" งานให้เป็นชิ้นเล็ก ๆ หลาย ๆ ชิ้นแยกออกจากกันให้ได้ อีกวิธีหนึ่งคือ ยึดถือการหั่นแบบแบ่งการคำนวณเป็นหลัก ซึ่งถ้าใช้ LINDA ก็เขียนง่ายทั้งสองวิธี จริง ๆ แล้วกรรมวิธีทั้งสองนั้นจะว่าไปก็มีส่วนคล้ายคลึงกันอยู่ (ก็หั่นเหมือน ๆ กัน) ฉะนั้นจะเลือกใช้วิธีไหนย่อมขึ้นอยู่กับฝีมือการมองหรือบางครั้งจะเล่นถูกผสมเลยยังได้

การ "หั่น" ข้อมูล

ในการเขียนโปรแกรมขนานโดยยึดการ "หั่น" แบ่งข้อมูล (Data decomposition) เป็นหลัก ข้อมูลจะถูกแบ่งเป็นชิ้นเล็กชิ้นน้อย แล้วจึงเอาไปคำนวณของใครของมัน ระบบนี้ใช้การได้ดีเมื่อมีข้อมูลจำนวนหนึ่งหน้าตาคล้าย ๆ กัน ต้องทำการคำนวณเหมือนกัน ตัวอย่างที่ง่ายที่สุดคือ ตารางสูตรคูณ

ตัวตั้ง	ตัวคูณ	ผลลัพธ์
1	2	2
2	2	4
3	2	6
4	2	8
5	2	10
6	2	12
7	2	14
8	2	16

ถ้าคำนวณทีละจำนวนก็จะต้องใช้เวลา 8 เท่า แต่ถ้าคำนวณครั้งละ 4 จำนวนจะใช้เวลา 2 เท่า และถ้าใช้การคำนวณพร้อมกันทีเดียว 8 จำนวนจะใช้เวลา 1 เท่า

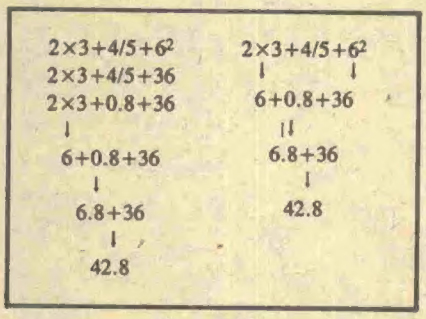
หรือตัวอย่างชนิดยาก ได้แก่ การคำนวณเพิ่มความคมชัดหรือเพิ่มความเนียนของภาพ ซึ่งจะต้องแบ่งรูปภาพเป็นชิ้นสี่เหลี่ยมเล็ก ๆ แล้วคำนวณทีละช่องได้เช่นกัน ปกติการตัดแบ่งข้อมูลนั้นจะมีเจ้าพ่ออยู่หนึ่งงานเป็นคนทำให้ก่อนแล้วแจกงานเหล่านี้ไปยังลิ่วล้อทั้งหลายเพื่อทำการคำนวณ ระบบเจ้าพ่อ-ลิ่วล้อ (Master-Worker) บางคนเรียกว่า "ไร่ปฏิจักร" (Processor farm)

ใน LINDA นั้นการติดต่อสื่อสารกันระหว่างงานย่อยจะใช้การส่งกลุ่มก้อนข้อมูล (tuple) ไปมาซึ่งกันและกัน งานเจ้าพ่อ (Master) จะเป็นผู้จัดแบ่งข้อมูลเป็นชิ้นเล็กชิ้นน้อย จากนั้นจะส่งออกไปยังงานลิ่วล้อ (Worker) ในรูปของกลุ่มก้อนข้อมูล บรรดาลิ่วล้อทั้งหลายก็จะเก็บข้อมูลเข้ามาทำการคำนวณ แล้วสร้างผลลัพธ์ในรูปกลุ่มก้อนข้อมูลส่งกลับมาให้งานเจ้าพ่อเพื่อรวบรวมเป็นผลงานสมบูรณ์แบบ การจัดการรอจังหวะเหมาะทำได้โดยการที่ให้งานย่อยหยุดรอถ้าข้อมูลที่ต้องการยังไม่มา ข้อดีของ LINDA ก็คือการใช้ระบบส่งกลุ่มก้อนข้อมูลทำให้ไม่จำกัดว่าเครื่องคอมพิวเตอร์จริง ๆ จะมีหน่วยประมวลผลเท่าไร คุณสามารถเขียนโปรแกรมเดียวทดลองกับเครื่องรุ่นเก่า (ที่มี LINDA) จนกว่าจะพอใจว่าไม่มีที่ผิดพลาด จากนั้นก็ย้ายโปรแกรมนี้ไปใช้กับเครื่องรุ่นใหม่ได้เลย

การ "หั่น" การคำนวณ

งานบางชนิดมีข้อมูลไม่มากแต่วิธีการคำนวณสลับซับซ้อน เปรียบเทียบกับวิชาปู้ลิ้ม สมมุติว่าเพลงกระบี่หนึ่งเพลงมี 64 กระบวน หนึ่งกระบวนมี 16 ท่า หนึ่งท่ามีความพลิกแพลง 7 อย่าง (งง !) หรืออย่างเช่น การประกอบรถยนต์คันเดียวต้องทำงานหลายสิบอย่าง เช่น เชื่อม พ่นสี ติดตั้งเครื่องยนต์ ตบแต่งภายใน ฯลฯ ฉะนั้นการแบ่งงานไปให้คน (หรือโปรเซสเซอร์) หลาย ๆ คนช่วยกันทำคนละอย่าง ทำเหมือนสายการผลิตแบ่งงานเป็นขั้น ๆ ทำงานขั้นที่หนึ่งเสร็จก็ส่งไปให้ขั้นที่สองทำ เมื่อขั้นที่หนึ่งว่าง

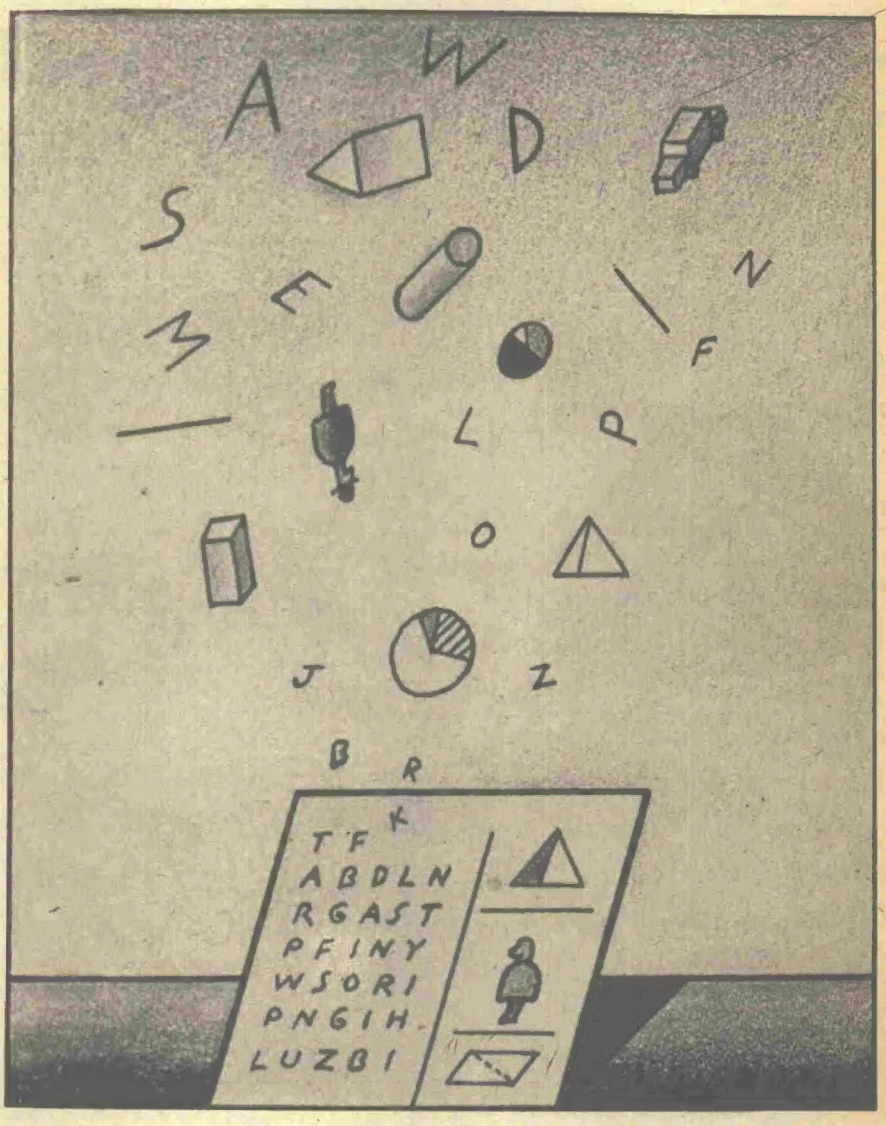
แล้วค่อยรับงานใหม่เข้ามาอย่างต่อเนื่อง ทีนี้ถ้าการแบ่งงานไม่ดีอาจมีบางขั้นตอนทำงานหนักกว่าปกติ ระบบสายการผลิตจะต้องคอยแผนกที่ทำเสร็จช้า (bottle-neck) ทางแก้ก็คือเพิ่มจำนวนคนไปช่วยตรงคอคอดนั้นเพื่อแบ่งเบาภาระ



ถ้าคำนวณทีละอย่างต้องใช้เวลา 5 เท่า ในขณะที่ถ้าใช้ระบบสายการผลิตจะใช้เวลา 3 เท่า

เพิ่มพลัง จริงเหรอ?

ในหลักทฤษฎีถ้ามีการเพิ่มหน่วยประมวลผลเข้าไป 1 ตัว เครื่องคอมพิวเตอร์ควรจะเพิ่มพลังขึ้น 1 เท่า เพิ่มเข้าไป 2 ตัวเท่ากับเพิ่มพลังเป็น 2 เท่า แต่ความจริงแล้วการเพิ่มหน่วยประมวลผลเข้าไปมาก ๆ ไม่อาจเพิ่มขีดความสามารถได้สักเท่าไรเลย ทั้งนี้เพราะเมื่อมีหน่วยประมวลผลหลายตัวกลับกลายเป็นว่าเริ่มมีเรื่องที่จะต้องควบคุมกำกับดูแลมากขึ้น ไหนจะเรื่องการติดต่อสื่อสาร การส่งถ่ายข้อมูล การจัดลำดับงาน และอื่น ๆ อีกจิปาถะ



การจะหาจุดเหมาะสม (Optimum) ขึ้นอยู่กับฝีมือของแต่ละคน LINDA นั้นได้ให้ขีดความสามารถในอันที่จะปรับแต่งโปรแกรมให้เข้าสู่จุดเหมาะสมได้ดังใจ แต่คุณยังจะต้องเป็นผู้ค้นหาจุดเหมาะสมดังกล่าวเอง ข้อสำคัญอันดับสุดท้ายคือ LINDA จะไม่ขึ้นกับเครื่องใดเครื่องหนึ่งโดยเฉพาะ โปรแกรมของคุณสามารถโอนย้ายจากเครื่องหนึ่งไปยังอีกเครื่องหนึ่งได้ โดยไม่ต้องกลัวว่าโปรแกรมที่เขียนใหม่จะล้าสมัยไปตามเครื่องรุ่นเก่า ต่อแต่นี้ไปอาวุธลับชนิดใหม่ล้วนอยู่ในอุ้งหัตถ์ของคุณ !